Generating Sentences from Data

Claire Gardent
(Joint work with Shashi Narayan and Bikash Gyawali)
CNRS/LORIA, Nancy

February 20, 2014
KU Leuven, Belgium

Outline

Generating Sentences from Dependency Trees

- The Generation Challenge Surface Realisation (SR) Task
- Feature-Based Lexicalised Tree Adjoining Grammar (FB-LTAG)
- A Structure-Driven Lexicalist Generation Algorithm
- Evaluation and results

Generating Sentence from Knowledge Bases

- The KBGen challenge
- Grammar Extraction
- Bottom-Up Lexicalist Generation
- Evaluation and Results

Generating Sentences from Dependency Trees

Claire Gardent
(Joint work with Shashi Narayan)

February 14, 2014

A TD/BU algorithm for Grammar Based Surface Realisation

What is Surface Realisation?

Structured Input from the Generation Challenge Surface Realisation Task

Feature-Based Tree Adjoining Grammar

The Algorithm

Evaluation and Results

S. Narayan and C. Gardent
Structure-Driven Lexicalist Generation
Proceedings of *COLING 2012*, pp 2027 - 2041, Mumbai, India

What is Surface Realisation?

SR maps INPUT DATA to SENTENCES

The input data can be more syntactic or more semantic; a tree or a graph:

- Dependency trees (SR Task)
- OWL triples
- First Order Logic (FOL) Formulae
- Flat semantics (MRSs)
- ...

$\exists x.(Man(x) \wedge \exists y.(Apple(y) \wedge eat(e,x,y) \wedge now(e)))$
$\Rightarrow$ *A man eats an apple*

The SR Shared Task Input Representations

Surface Realisation Task organised by the Generation Challenges 2011

Input derived from the PennTreebank

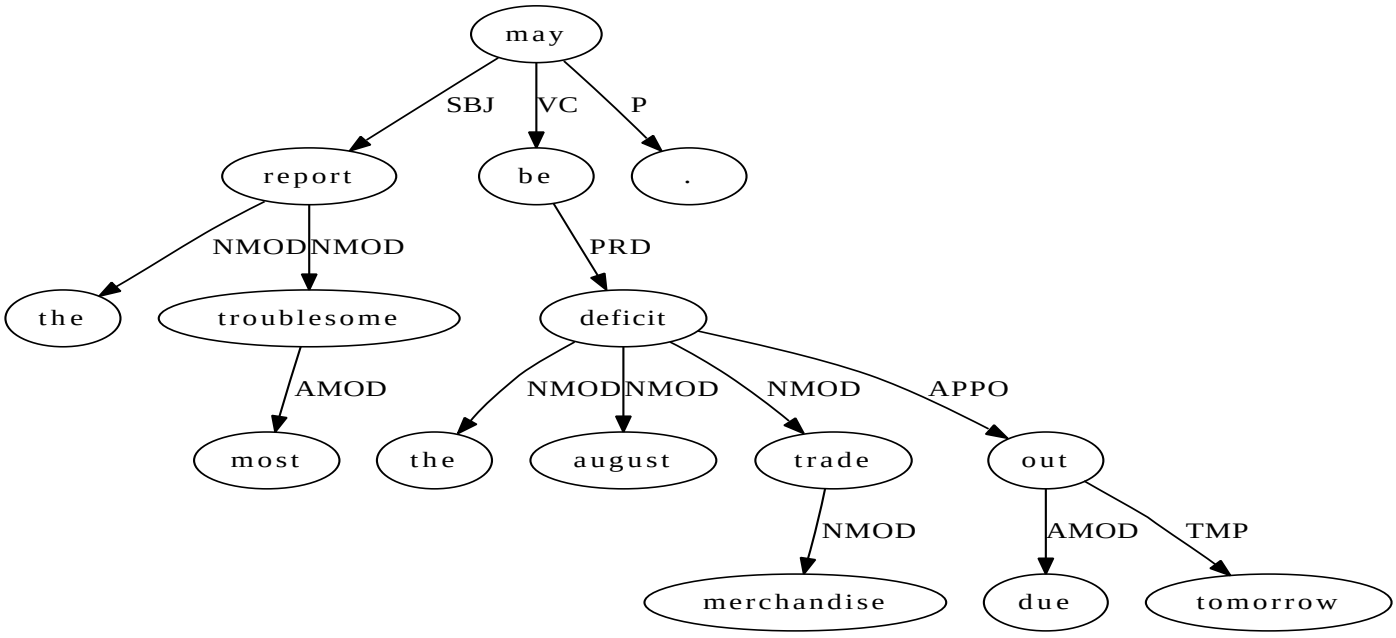
Shallow dependency structures

- Unordered trees
- Edges are labelled with syntactic functions
- Nodes labelled with lemmas, part of speech tags and partial morphosyntactic information

All words of the original sentence are represented by a node in the tree

Example Input

The most troublesome report may be the August merchandise trade deficit due out tomorrow

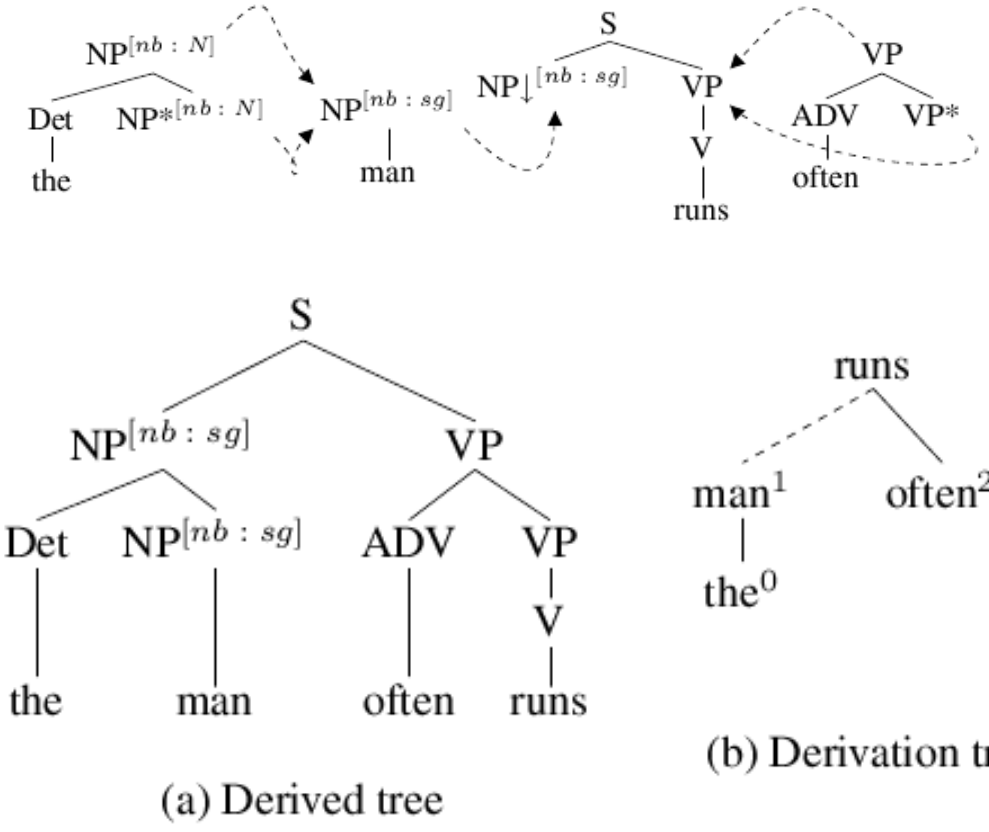


Grammar

Feature-Based Lexicalised Tree Adjoining Grammar (FB-LTAG)

- A set of trees, lexicalised with one or more words and decorated with feature structures
- 2 combining operations: substitution and adjunction
- XMG Reimplementation of XTAG (large coverage of English)

Example FB-LTAG



Grammar-Based Surface Realisation Algorithms

Two main approaches

Head-Driven algorithm

Used for recursively structured input data e.g., logical formulae
Use input structure to guide the search
Cons: Logical Form Equivalence Problem

Lexicalist

Used for unstructured input data (e.g., MRS formula)
Selects lexical entries bottom-up from the input semantic literals
Cons: Computationally expensive (Unordered input, Lexical ambiguity, Intersective modifiers)

Structure-Driven Lexicalist Surface Realisation

Combines techniques and ideas from the head-driven and the lexicalist approach.

- Select grammar rules bottom up for each input tree node (Lexicalist)
- Uses the structure of the input to guide the search and prune the search space (Structure Driven)

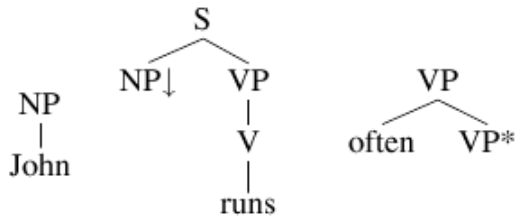
Integrates various optimisations previously proposed for parsing/generation

Parallelised

Structure-Driven Lexicalist Generation

- FB-LTAG **converted to FB-RTG** to construct derivation rather than derived trees
  (Koller and Striegnitz, 2002; Gardent and Perez-Beltrachini 2010)
- Top-down filter using the structure of the input
  (**Head-Driven** algorithm, Shieber et al. 1990)
- Bottom-up **polarity filter** on local input trees.
  (Bonfante 2004; Gardent and Kow 2007).
- **Language model** used to prune competing intermediate structures
  (Bangalore and Rambow 2000; White 2004)
- **Parallelism** used to explore the possible completions of the top-down predictions simultaneously rather than sequentially.

Converting a TAG to an RTG



r1.	NP_S	$\rightarrow$	$john(NP_A)$
r2.	S_S	$\rightarrow$	$runs(S_A\ NP_S\ VP_A\ V_A)$
r3.	VP_A	$\rightarrow$	$often(VP_A)$
r4.	NP_A	$\rightarrow$	ϵ
r5.	S_A	$\rightarrow$	ϵ
r6.	V_A	$\rightarrow$	ϵ
r7.	VP_A	$\rightarrow$	ϵ

The Algorithm

Starts from the root node of the input tree

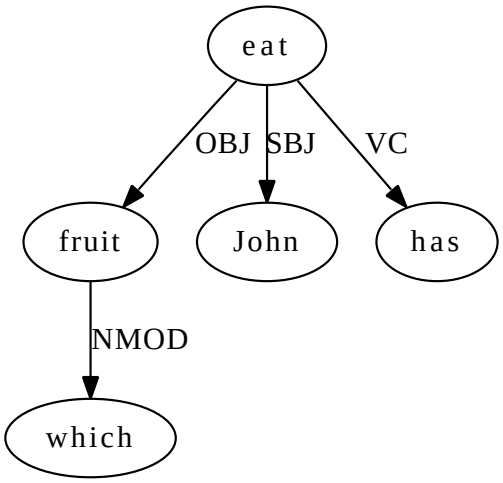
Processes all children nodes *in parallel* spreading lexical selection constraints *top-down* and combining FB-RTG rules *bottom-up*

4 main steps

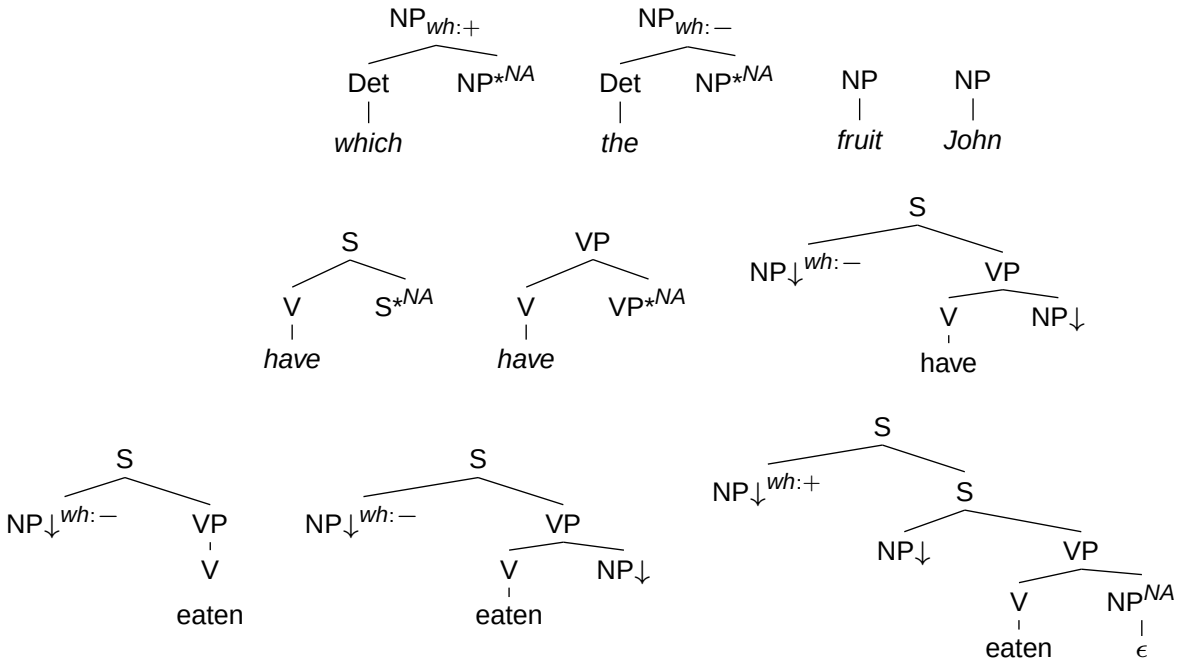
- Bottom-Up Lexical Selection and Top-Down Filtering
- Bottom-Up Local Polarity Filtering
- Bottom-Up Generation
- N-Gram Filtering

Example

Input Dependency Tree



An FB-LTAG


NP$_{wh:+}$
Det
NP*^{NA}
which
NP$_{wh:-}$
Det
NP*^{NA}
the
NP
fruit
NP
John
S
V
S*^{NA}
have
VP
V
VP*^{NA}
have
S
NP↓$^{wh:-}$
VP
V
NP↓
have
S
NP↓$^{wh:-}$
VP
V
eaten
S
NP↓$^{wh:-}$
VP
V
NP↓
eaten
S
NP↓$^{wh:+}$
S
NP↓
VP
V
NP^{NA}
eaten
ϵ

... and the corresponding FB-RTG

$$
\begin{array}{lcl}
NP_A^{[t:T]} & \rightarrow & \mathit{which}(NP_A^{[t:T,b:[wh:+]]}) \\
NP_A^{[t:T]} & \rightarrow & \mathit{the}(NP_A^{[t:T,b:[wh:-]]}) \\
NP_S^{[t:T]} & \rightarrow & \mathit{fruit}(NP_A^{[t:T]}) \\
NP_S^{[t:T]} & \rightarrow & \mathit{John}(NP_A^{[t:T]}) \\
S_A^{[t:T]} & \rightarrow & \mathit{have}(S_A^{[t:T]}) \\
VP_A^{[t:T]} & \rightarrow & \mathit{have}(VP_A^{[t:T]}) \\
S_S^{[t:T,b:B]} & \rightarrow & \mathit{have}(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S) \\
S_S^{[t:T,b:B]} & \rightarrow & \mathit{eat}(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A) \\
S_S^{[t:T,b:B]} & \rightarrow & \mathit{eat}(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S) \\
S_S^{[t:T,b:B]} & \rightarrow & \mathit{eat}(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:+]]}\ S_A\ NP_S\ VP_A) \\
X_A^{[t:T,b:T]} & \rightarrow & \epsilon
\end{array}
$$

Bottom-Up Lexical Selection and Top-Down Filtering

Lexical Selection: for each input node *n* with lemma *w*, selects all FB-RTG rules which can be lexicalised by *w*

Top-Down Filtering: Only keep those rules whose left-hand side category occurs at least once in the right-hand side of the rules selected by the parent node.

Example Top-Down Filtering

Rule selection for *eat*:

$$
\begin{array}{lll}
S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A) \\
S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S) \\
S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:+]]}\ S_A\ NP_S\ VP_A)
\end{array}
$$

Rule selection and filtering for *has*:

$$
\begin{array}{lll}
\surd & S_A^{[t:T]} & \rightarrow\ have(S_A^{[t:T]}) \\
\surd & VP_A^{[t:T]} & \rightarrow\ have(VP_A^{[t:T]}) \\
\times & S_S^{[t:T,b:B]} & \rightarrow\ have(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S)
\end{array}
$$

Bottom-Up Local Polarity Filtering

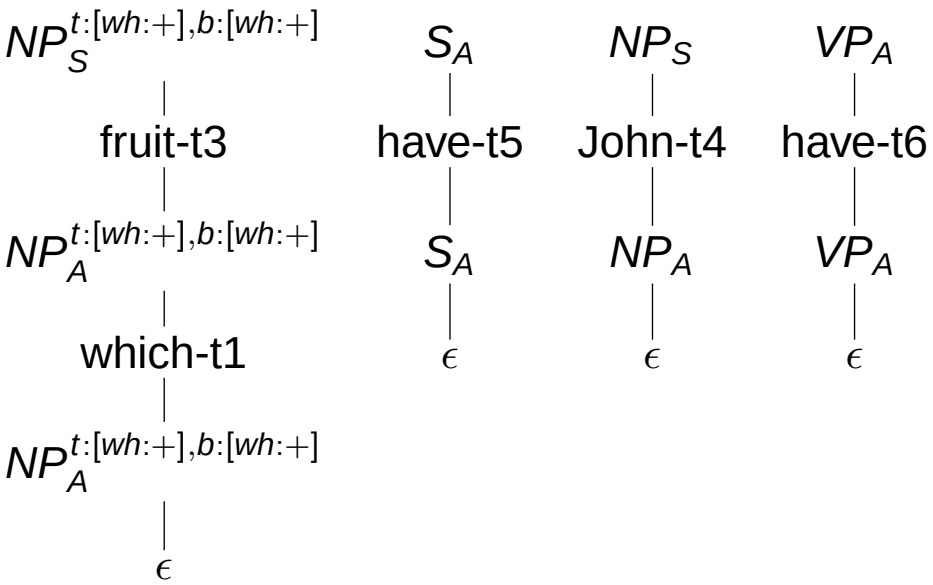
Global Polarity Filtering (Gardent and Kow 2005) filters out

- Sets of rules which cover the input
- but cannot possibly lead to a valid derivation
- either because a substitution node cannot be filled
- or because a root node fails to have a matching substitution site

Local (Structure-Driven) Polarity Filtering: on each local tree

Example of Local Polarity Filtering

$$
\begin{array}{lcll}
\times & S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A) \\
\surd & S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S) \\
\surd & S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:+]]}\ S_A\ NP_S\ VP_A)
\end{array}
$$



Bottom-Up Generation and N-Gram filtering

For each local tree in the input, the rule sets passing the local polarity filter are tried out for combination.

Only *the n best scoring n-grams* let through after each bottom-up generation step are kept.
The language model helps finding the most likely ordering of modifiers.

Evaluation and Results

Test data: The SR Data

- Dependency trees derived from the Penn Treebank
- 26 725 inputs
- Average (maximum) word length: 22 (134)
- Average (maximum) branching degree: 4 (18)

Algorithms compared:

- Baseline: A strictly top-down algorithm
  (No time information available for systems participating in SR Task, only coverage and BLEU)
- SEQ: The SDL algorithm without parallelism
- PAR: The SDL algorithm with parallelism

Evaluation Focus: Efficiency (Time)

Evaluation and Results

	Sentences (Length L)			
	$S(0-5)$	$S(6-10)$	$S(11-20)$	$S(All)$
	Total	Total	Total	Total
	1084	2232	5705	13661
BL	**0.85**	10.90	110.07	–
SEQ	1.49	2.84	4.36	4.52
PAR	**1.53**	2.56	2.66	**2.57**

- Maximum arity = 3. Else BL times out.
- Many time out for BL on input longer than 10
- For short sentences (0-5), BL outperforms SDL
- For sentences with more than 5 words, SDL increasingly outperforms BL

Branching factor and Parallelism

	Sentences (Arity)			
	$S(1)$ Total 190	$S(3)$ Total 3619	$S(5)$ Total 2910	$S(6)$ Total 1093
SEQ	**0.89**	3.65	5.24	8.20
PAR	**0.97**	2.63	2.86	3.09

The impact of parallelism increases with the branching factor.

Coverage and BLEU score

Coverage: 81.74%

- No robustness mechanism added.

BLEU score: 0.73 (for covered data)

- No ranking module
- Best statistical system in SR Task: 0.88
- Best symbolic system in SR Task: 0.37

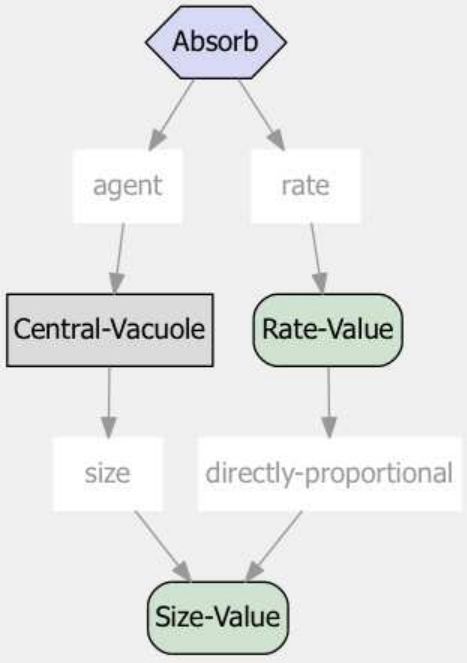
Generating Sentences from Knowledge Bases

Claire Gardent
(Joint work with Bikash Gyawali)

B. Gyawali and C. Gardent
LOR-KBGEN, A Hybrid Approach to Generating from the KBGen Knowledge-Base
Proceedings of *ENLG 2013*, pp. 204 - 205, Sofia, Bulgaria

The KBGen Task

Given a set of relations selected from the AURA knowledge base, generate complex sentences that are grammatical and fluent in English.



The rate of absorption of a central vacuole is directly proportional to the size of the vacuole.

Data provided to the participants

- 207 input/outpupt pairs for training
- 72 input for testing (automatic and human-based evaluation)
- inputs in graph form
- corresponding output sentences
- lexicon

Example Input/Output Pair

The function of a gated channel is to release particles from the endoplasmic reticulum

```
:TRIPLES (
(|Release-Of-Calcium646| |object| |Particle-In-Motion64582|)
(|Release-Of-Calcium646| |base| |Endoplasmic-Reticulum64603|)
(|Gated-Channel64605|  |has-function||Release-Of-Calcium646|)
(|Release-Of-Calcium646|  |agent| |Gated-Channel64605|))
:INSTANCE-TYPES
(|Particle-In-Motion64582| |instance-of| |Particle-In-Motion|)
(|Endoplasmic-Reticulum64603| |instance-of| |Endoplasmic-Retic
(|Gated-Channel64605| |instance-of| |Gated-Channel|)
 |Release-Of-Calcium646| |instance-of| |Release-Of-Calcium|))
:ROOT-TYPES (
(|Release-Of-Calcium646| |instance-of| |Event|)
(|Particle-In-Motion64582| |instance-of| |Entity|)
(|Endoplasmic-Reticulum64603| |instance-of| |Entity|)
(|Gated-Channel64605| |instance-of| |Entity|)))
```

Lexical resources provided

- For events: a verb, its nominalization and different word forms (Concept, 3sg, base form, past participle, nominalization):

  `Release-Of-Calcium: releases,release,released,release`

- For entities: a noun or noun phrase, and its plural form (Concept, singular noun, plural):

  `Particle-In-Motion:  a molecule in motion, molecules in motion`

The LOR-KBGEN approach

Parse the training sentences

Induce an FB-LTAG with semantics from (input,output) pairs using the parse trees

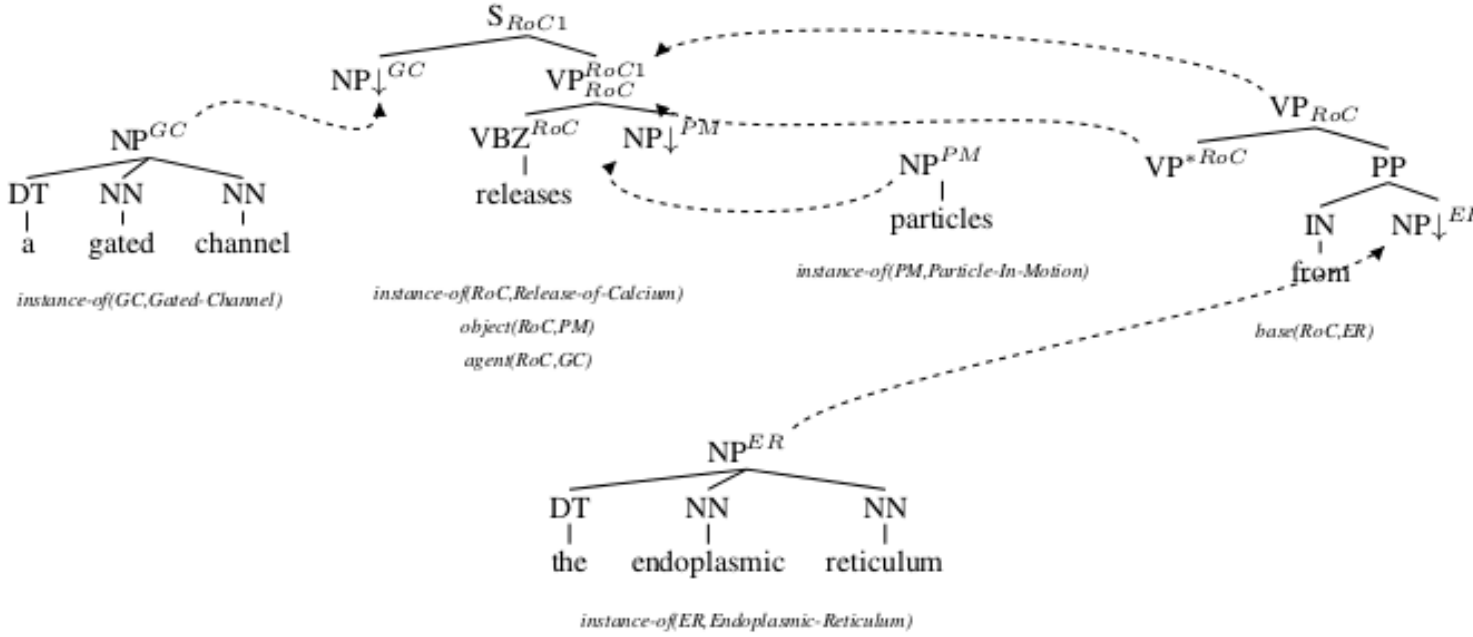
Generate with an existing lexicalist generator

FB-LTAG with Semantics

FB-LTAG: trees decorated with feature structures and combined using adjunction and substitution

Each tree is associated with a semantic schema and (unification) variables are shared between the tree feature structures and the semantic schema

Example FB-LTAG with Semantics



Grammar Induction

Align entity and event variables with words

Project these variables up the parse tree to the corresponing maximum projection

Extract subtrees from the parse tree such that each subtrees describes a coherent syntactic/semantic unit

Example Alignment

The function of a (gated channel, `Gated-Channel64605`) is to (release, `Release-Of-Calcium646`) (particles, `Particle-In-Motion64582`) from the (endoplasmic reticulum, `Endoplasmic-Reticulum64603`)

Grammar Induction (I)

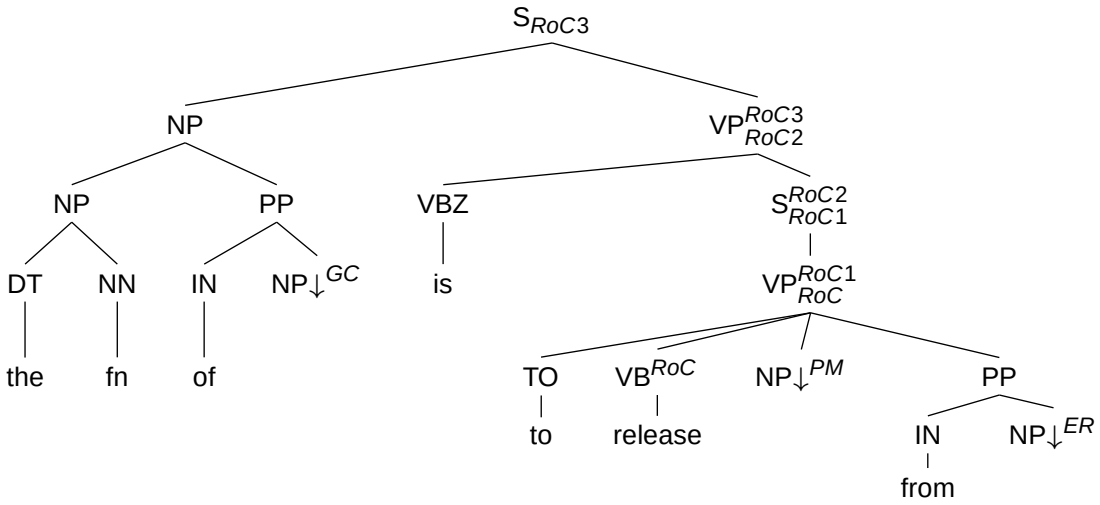
Variable projection

- A variable aligned with a noun is projected to the NP level or to the immediately dominating PP if it occurs in the subtree dominated by the leftmost daughter of that PP.
- A variable aligned with a verb is projected to the first S node immediately dominating that verb or, in the case of a predicative sentence, to the root of that sentence

Tree Extraction

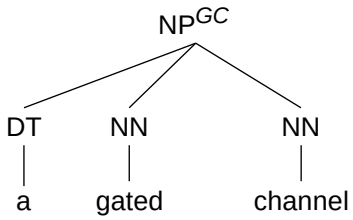
- NP trees: subtrees whose root node are indexed with an entity variable
- S and PP trees: subtrees capturing relations between variables. Minimal tree containing all and only the dependent variables of a variable

Example Extraction: S tree

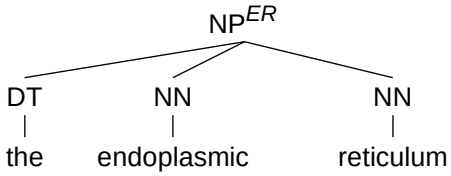


instance-of(RoC,Release-of-Calcium)
object(RoC,PM)
base(RoC,ER)
has-function(GC,RoC)
agent(RoC,GC)

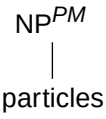
Example Extraction: NP trees



instance-of(GC,Gated-Channel)



instance-of(ER,Endoplasmic-Reticulum)



instance-of(PM,Particle-In-Motion)

Generation

Sentences are generated using the GenI surface realiser.

- Given input semantics ϕ, select all trees in the grammar whose semantics subsumes ϕ.
- Combines the resulting trees using substitution and adjunction
- Generated sentences are derived sentences whose associated semantics is ϕ

C. Gardent and E. Kow

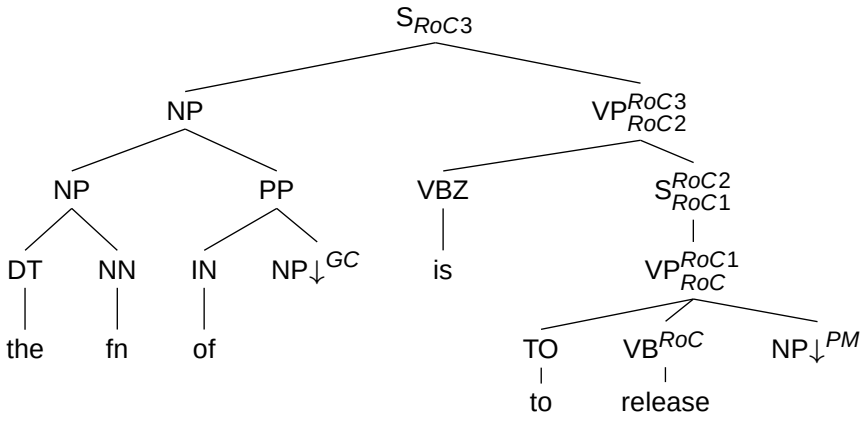
A symbolic approach to Near-Deterministic Surface Realisation using Tree Adjoining Grammar.
Proceedings of *ACL 2007*, Prague, Tcheck Republic

Handing Unseen Configurations

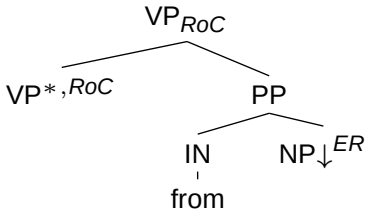
The extracted grammar overfits the data. To reduce overfitting, we generalise the grammar by extracting :

- S subtrees describing transitive verbs (with the corresponding semantics)
- VP or PP subtrees describing modifiers (with the corresponding semantics)

Example Generalisations



instance-of(RoC,Release-of-Calcium)
object(RoC,PM)
has-function(GC,RoC)
agent(RoC,GC)



base(RoC,ER)

Evaluation and Results

Evaluation setup

- 72 input from KBGEN
- 3 competing systems
- Automatic (BLEU) and Human-Based Evaluation
- Three configurations for our approach
  - BASE: without grammar expansion
  - MANEXP: with manual grammar expansion
  - AUTEXP: with automated grammar expansion

Automatic Evaluation

System	All	Covered	Coverage	# Trees
IMS	0.12	0.12	100%	
UDEL	0.32	0.32	100%	
Base	0.04	0.39	30.5%	371
ManExp	0.28	0.34	83 %	412
AutExp	0.29	0.29	100%	477

Figure: BLEU scores and Grammar Size (Number of Elementary TAG trees

Manual Evaluation

12 participants were asked to rate sentences along three dimensions:

- **fluency**: Is the text easy to read?
- **grammaticality**: Is the text grammatical ?
- **adequacy**: Does the meaning conveyed by the generated sentence correspond to the meaning conveyed by the reference sentence?

Online evaluation (LG-Eval toolkit)

Subjects used a sliding scale from -50 to +50

Latin Square Experimental Design was used to ensure that each evaluator sees the same number of output from each system and for each test set item.

Results

	Fluency		Grammaticality		Meaning Similarity	
System	Mean		Mean		Mean	
UDEL	4.36	B	4.48	B	3.69	A
AutExp	3.45	C	3.55	C	3.65	A
IMS	1.91	D	2.05	D	1.31	B

Systems are grouped by letters when there is no significant difference between them (significance level: $p < 0.05$, post-hoc Tukey test)

LOR-KBGEN ranks 2nd behind the symbolic, UDEL system and before the statistical IMS approach

Conclusion

TAG extended domain of locality and semantic principle:

> *TAG trees group together in a single structure a syntactic predicate and its arguments. Each elementary tree captures a single semantic unit*

Our grammar extraction approach supports the extraction of a grammar which respects those principes and enforces strong constraints on generated sentences.

Thanks ! Questions ?